

Word Ladder – II [\(View\)](#)

A **transformation sequence** from word `beginWord` to word `endWord` using a dictionary `wordList` is a sequence of words `beginWord -> s1 -> s2 -> ... -> sk` such that:

- Every adjacent pair of words differs by a single letter.
- Every `si` for `1 <= i <= k` is in `wordList`. Note that `beginWord` does not need to be in `wordList`.
- `sk == endWord`

Given two words, `beginWord` and `endWord`, and a dictionary `wordList`, return *all the **shortest transformation sequences** from `beginWord` to `endWord`, or an empty list if no such sequence exists. Each sequence should be returned as a list of the words `[beginWord, s1, s2, ..., sk]`.*

Example 1:

Input: `beginWord = "hit", endWord = "cog", wordList = ["hot","dot","dog","lot","log","cog"]`

Output: `[["hit","hot","dot","dog","cog"],["hit","hot","lot","log","cog"]]`

Explanation: There are 2 shortest transformation sequences:

"hit" -> "hot" -> "dot" -> "dog" -> "cog"

"hit" -> "hot" -> "lot" -> "log" -> "cog"

Example 2:

Input: `beginWord = "hit", endWord = "cog", wordList = ["hot","dot","dog","lot","log"]`

Output: `[]`

Explanation: The endWord "cog" is not in wordList, therefore there is no valid transformation sequence.

Constraints:

- `1 <= beginWord.length <= 5`
- `endWord.length == beginWord.length`
- `1 <= wordList.length <= 500`
- `wordList[i].length == beginWord.length`
- `beginWord`, `endWord`, and `wordList[i]` consist of lowercase English letters.
- `beginWord != endWord`
- All the words in `wordList` are **unique**.